(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENT- UND MARKENAMT

(12) **Offenlegungsschrift**

(10) **DE 199 47 986 A 1**

(51) Int. Cl.[7]:
H 04 L 9/00
G 06 F 17/60

(21) Aktenzeichen: 199 47 986.0
(22) Anmeldetag: 5. 10. 1999
(43) Offenlegungstag: 12. 4. 2001

DE 199 47 986 A 1

(71) Anmelder:
International Business Machines Corp., Armonk, N.Y., US

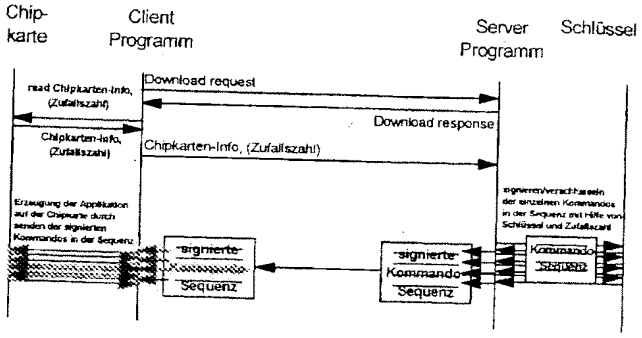
(74) Vertreter:
Klein, H., Rechtsanw., 70569 Stuttgart

(72) Erfinder:
Schäck, Thomas, 77855 Achern, DE; Hepper, Stefan A., 72074 Tübingen, DE

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) System und Verfahren zum Herunterladen von Anwendungsteilen auf eine Chipkarte

(57) Die vorliegende Erfindung beschreibt ein Verfahren zum Herunterladen von Anwendungsteilen, sogenannte On-Card-Anwendungsteile, von einem Server über einen Client auf eine Chipkarte, wobei Server und Client über ein verteiltes System, insbesondere Intranet oder Internet, miteinander kommunizieren. Die Vorteile der vorliegenden Erfindung liegen darin, dass das Herunterladen der Anwendungsteile in zwei Stufen erfolgt: Die erste Stufe erfolgt ausschließlich auf dem Server und stellt sicher, dass nicht jedes Kommando zum Herunterladen des Anwendungsteils einzeln über das Netz verschickt wird. Dies erfolgt mittels eines bandbreitoptimierten Protokolls, das die einzelnen Kommandos zum Herunterladen des Anwendungsteils zu einer Kommando-Sequenz zusammenfasst und als ein gesamtes Datenpaket über das Netz schickt. Dadurch wird die Zeit zum Herunterladen von Anwendungsteilen über das Netz reduziert. Jedes Kommando innerhalb der Kommando-Sequenz wird hierbei mit einer digitalen Signatur versehen und gegebenenfalls verschlüsselt. Damit wird sichergestellt, dass nur authentifizierte Kommandos von der Chipkarte akzeptiert werden. Damit erfüllt diese Erfindung sicherheitsrelevante Anforderungen beim Übertragen von Daten über verteilte Systeme, insbesondere über das Internet.
Die zweite Stufe erfolgt zwischen Client und Chipkarte und stellt sicher, dass die Datenpakete entpackt und einzeln zur Chipkarte gesandt werden. Alle sicherheitsrelevanten Schlüssel und Zertifikate sind auf dem ...



DE 199 47 986 A 1

Beschreibung

Die vorliegende Erfindung beschreibt ein System und Verfahren zum Herunterladen von Anwendungsteilen über verteilte Systeme auf Chipkarten, insbesondere auf Chipkarten, die sich bereits im Einsatz befinden.

Normalerweise werden Chipkarten mit vorbereiteten On-Card-Anwendungsteilen ausgeliefert.

Diese On-Card-Anwendungsteile ermöglichen die Kommunikation zwischen der Chipkarte und den Chipkartenanwendungen, den sogenannten Off-Card-Anwendungen, die auf einem Terminal, z. B. Server-System, installiert sind. Die Chipkarte, d. h der On-Card-Anwendungsteil, kommuniziert über ein Chipkartenlesegerät mit dieser Off-Card-Anwendung. Moderne Chipkarten, sogenannte Multifunktions-Chipkarten, wie z. B. Java Card oder Smart Cards für Windows, haben zusätzlich die Funktionalität, dass ON-Card-Anwendungsteile nachträglich, d. h nach Auslieferung der Chipkarten, auf die Chipkarte gebracht werden können (siehe **Fig.** 1). Hierbei erfolgt das Herunterladen der On-Card-Anwendungsteile vom Terminal über das Chipkartenlesegerät auf die Chipkarte.

Z. B. VISA hat eine Open Platform Spezifikation definiert, die die Kommandos zwischen Off-Card-Anwendung und dem On-Card-Anwendungsteil, dem On-Card-Interface und den Sicherheitsstandards beschreibt. OCF (OpenCard-Framework) und Microsoft's PC/SC auf der anderen Seite adressieren die Kommunikation zwischen Anwendung, Chipkartenlesegerät und Chipkarte.

Durch den verstärkten Einsatz von verteilten Systemen entsteht zunehmend das Bedürfnis, On-Card-Anwendungsteile über verteilte Systeme auf die Chipkarte zu laden. Die Risiken solcher Verfahren liegen auf der Hand. Das Netz unterliegt unterschiedlichen Auslastungen, so dass das Herunterladen je nach Auslastung viel Zeit in Anspruch nehmen kann. Ein weiterer wesentlicher Aspekt ist hier die Sicherheit. Alle Datenübertragen von dem Server über den Client auf die Chipkarte müssen gesichert sein. Es muß ein einfaches und sicheres Authentifizierungs- und Verschlüsselungsverfahren beim Herunterladen der Anwendungsteile sichergestellt zum Einsatz kommen, das den unterschiedlichen Auslastungen des Netzes Rechnung trägt.

Derzeit sind der Anmelderin jedoch keine Systeme bzw. Verfahren bekannt, die diese Einsatzmöglichkeit adressieren.

Es ist daher Aufgabe der vorliegenden Erfindung, ein System und Verfahren zum Herunterladen von Anwendungsteilen über verteilte Systeme auf eine Chipkarte sicherzustellen, das auf einfache Weise unter Berücksichtigung der erforderlichen sicherheitsrelevanten Überprüfungen erfolgt.

Diese Aufgabe wird durch die Merkmale der Ansprüche 1, 17, 18 und 20 gelöst. Vorteilhafte Ausführungsformen der vorliegenden Erfindung sind in den Unteransprüchen niedergelegt.

Die Vorteile der vorliegenden Erfindung liegen darin, dass das Herunterladen der Anwendungsteile in zwei Stufen erfolgt:

Die erste Stufe erfolgt ausschließlich auf dem Server und stellt sicher, dass nicht jedes Kommando zum Herunterladen des Anwendungsteils einzeln über das Netz verschickt wird. Dies erfolgt mittels eines optimierten Protokolls, das die einzelnen Kommandos zum Herunterladen des Anwendungsteils zu einer Kommando-Sequenz zusammenfasst und als ein Datenpaket über das Netz schickt. Dadurch wird die Zeit zum Herunterladen von Anwendungsteilen über das Netz reduziert. Jedes Kommando innerhalb der Kommando-Sequenz wird mit einer digitale Signatur versehen und gegebenfalls verschlüsselt. Damit wird sichergestellt, dass nur

authentifizierte Kommandos von der Chipkarte akzeptiert werden.

Damit erfüllt diese Erfindung sicherheitsrelevante Anforderungen beim Übertragen von Daten über verteilte Systeme, insbesondere das Internet.

Die zweite Stufe erfolgt zwischen Client und Chipkarte und stellt sicher, dass die Datenpakete entpackt und einzeln zur Chipkarte gesandt werden.

Alle sicherheitsrelevanten Schlüssel und Zertifikate werden auf dem sicheren Server abgelegt. Die Kommunikation zwischen Client und Server erfolgt vorzugsweise über SSL (Secure Sockets Layer) als Übertragungsprotokoll. Der Missbrauch des erfinderischen Systems/Verfahrens wird hierdurch wesentlich erschwert.

Die vorliegende Erfindung wird anhand bevorzugter Ausführungsbeispiele anhand Figuren näher beschrieben, wobei

Fig. 1 den Stand der Technik der Kommunikation zwischen Off-Card-Anwendung und On-Card-Anwendungsteil zeigt,

Fig. 2 eine verteile Kommunikationsarchitektur zeigt, die der vorliegenden Erfindung zugrunde liegt,

Fig. 3 die erfinderischen Schritte beim Herunterladen von On-Card-Anwendungsteilen von einem Server über ein Netz auf eine Chipkarte zeigt,

Fig. 4 die erfinderische Architektur nach **Fig.** 3 in einer Java-Implementierung zeigt,

Fig. 5 die erfinderischen Schritte beim Herunterladen von On-Card-Anwendungsteilen von einem Server über Netz auf eine Chipkarte in einer Java-Implementierung zeigt.

Fig. 1 zeigt den Stand der Technik beim Herunterladen von On-Card-Anwendungsteilen von einem Terminal auf die Chipkarte und die Kommunikation zwischen On-Card-Anwendungsteil und Off-Card-Anwendung. Im Stand der Technik bestehen Chipkartenkartenanwendungen aus einer Off-Card-Anwendung, die auf einem Terminal abgespeichert ist, und einem On-Card-Anwendungsteil, der auf der Chipkarte im nichtflüchtigen Speicher abgelegt ist (siehe **Fig.** 1). Das Terminal besteht aus einem Datenverarbeitungsgerät mit einem Chipkartenleser und der entsprechenden Treibersoftware für den Chipkartenleser. Der On-Card Anwendungsteil kommuniziert mit der Off-Card-Anwendung über mehrere Schichten. Schicht 1 definiert das physikalische Transportprotokoll. Schicht 2 setzt auf dieses Protokoll ein logisches, byte-orientiertes Protokoll. Schicht 3 bildet höhere Programmiersprache auf Schicht 2 ab. Ein Beispiel für die Schicht 1 ist das Protokoll T = 0, T = 1(ISO/IEC7816-3), Schicht 2 APDU Protokoll(ISO/7816-4), Schicht 3 OCF(Open Card Framework) oder PCSC().

Normalerweise wird der On-Card-Anwendungsteil über eine Ladeapplikation, die im Terminal läuft, auf die Chipkarte übertragen. Hierbei werden geeignete Kommandos der Chipkarte verwendet (z. B. bei File-orientiertenchipkarten "CREATE-und UPDATE Kommandos"). Derzeit gibt es noch keine bekannte Lösung zur Übertragung von On-Card-Anwendungsteilen über verteilte Systeme auf die Chipkarte.

Fig. 2 zeigt die erfinderische Architektur der vorliegenden Erfindung.

Die erfinderische Architektur basiert auf einer Client/Server-Architektur. Der Client kommuniziert mit dem Server über ein Netzwerk, z. B. dem Internet oder Intranet. Der Client ist mit einem Chipkartenleser verbunden und nur der Server hat Zugriff auf die geheimen Schlüssel, die zum Herunterladen von On-Card-Anwendungsteilen auf die Chipkarte erforderlich sind. Die Schlüssel können entweder auf dem Server selbst oder auf einem weiteren System abgelegt sein, auf das der Server Zugriff hat. Die Chipkarte ist gegen unberechtigtes Herunterladen von On-Card-Anwendungsteilen derart gesichert, dass sie Kommandos nur dann akzep-

tiert, wenn sie mit den richtigen Schlüsseln signiert und/oder verschlüsselt sind. Auf dem Client muß zur Laufzeit ein Programm existieren, das sowohl mit der Chipkarte als auch dem Server kommuniziert und das ein Protokoll implementiert, das von der jeweiligen Chipkarte abhängt.

In diesem Protokoll ist festgelegt, wann welche Nachrichten mit der Chipkarte und dem Server ausgetauscht werden müssen. Auf dem Server muß zur Laufzeit ein Programm existieren, das mit dem Client kommuniziert und bei Bedarf die dem Server zugänglichen Schlüssel verwendet, und das ein Protokoll implementiert, das festlegt, wann welche Nachrichten mit dem Client ausgetauscht werden müssen und wann welche Schlüssel verwendet werden müssen. Bei der Chipkarte handelt es sich um gängige Chipkarten (z. B. Java Card oder File orientierte Chipkarten), die für die vorliegende Erfindung nicht angepaßt werden müssen.

Fig. 3 zeigt die erfinderischen Schritte beim Herunterladen von On-Card-Anwendungsteilen von einem Server über ein Netz auf eine Chipkarte.

Der Client stellt eine Kommunikation zur Chipkarte und zum Server her.

Der Client stellt einen Request an den Server, dass ein On-Card-Anwendungsteil (Anwendungsteil A) auf die Chipkarte gebracht werden soll. Client und Server kommunizieren vorzugsweise über TCP/IP oder HTTP.

Der Server sendet eine Response an den Client, mit der Aufforderung, die Chipkartenidentifikationsdaten und gegebenenfalls eine Zufallszahl zur Authentifikation zu senden. Chipkartenidentifikationsdaten beinhalten zumindest Daten über den Chipkartentyp und die Chipkartennummer.

Der Client empfängt die Response vom Server und sendet geeignete Kommando-APPUs an die Chipkarte, um die Chipkartenidentifikationsdaten und gegebenenfalls eine Zufallszahl zu erhalten. Die Chipkartenidentifikationsdaten sind im nichtflüchtigen der Chipkarte abgelegt und können durch geeignete Kommandos ausgelesen werden. Die Chipkarte empfängt die Kommandos und gibt die Chipkartenidentifikationsdaten und gegebenfalls die Zufallszahl an den Client zurück. Der Client sendet diese Daten in einem Request an den Server.

Der Server empfängt den Request und wertet die Chipkartenidentifikationsdaten aus, um herauszufinden, welche Schlüssel verwendet werden müssen bzw. um die erforderlichen Schlüssel aus Master-Schlüsseln abzuleiten, um den Anwendungsteil A herunterladen zu können. Die Schlüssel werden verwendet, um eine Kommando-Sequenz für das Herunterladen der Anwendung A von dem Server auf die Chipkarte vorzubereiten. Diese Kommando-Sequenz bewirkt auf der Chipkarte die Erzeugung der Anwendung A. Die Kommando-Sequenz ist eine vordefinierte Sequenz, die nichtflüchtigen Speicherbereich des Servers für eine bestimmte Anwendung abgelegt ist. Eine weitere Ausführungsform ist, dass die Kommando-Sequenz ganz oder teilweise mit Hilfe eines Programms auf dem Server erzeugt wird. Dies wird vorzugsweise dann angewendet werden, wenn mit Hilfe der Kommando-Sequenz auch kartenindividuelle Daten mit in den On-Card-Anwendungsteil integriert werden sollen. Vorzugsweise wird jedes Kommando innerhalb der Sequenz mit Hilfe des Schlüssels (SessionKeys) signiert und gegenfalls verschlüsselt. Dies kann zum Beispiel dadurch erfolgen, dass das erste Kommando innerhalb der Sequenz mit Hilfe der Zufallszahl und des richtigen Schlüssels mit einem MAC (message authentification code) versehen wird und alle nachfolgenden Kommandos mit Hilfe des MACs des vorangehenden Kommandos und des richtigen Schlüssels mit einem MAC versehen werden. Die Sequenz mit den signierten und gegebenenfalls verschlüsselten Kommandos wird zum Client gesendet.

Der Client empfängt die Response mit der Kommando-Sequenz und sendet die Kommandos nacheinander zur Chipkarte. Die Chipkarte überprüft die Signatur und führt die Kommandos nur bei Richtigkeit aus.

Fig. 4 zeigt die erfinderische Architektur nach **Fig.** 3 in einer Java-Implementierung.

Auf dem Client läuft ein Web-Browser mit dessen Hilfe der Benutzer zur Web-Seite des Servers navigieren kann. Auf der Web-Seite des Servers befindet sich das Applet, das das in **Fig.** 3 beschriebene Client-Programm implementiert. Dieses Applet wird beim Anzeigen des Web-Seite vom Server in den Browser heruntergeladen. Das Applet baut eine Kommunikationsverbindung zu einem Servlet auf dem Server auf. Das Servlet besitzt die Funktionalität des Server-Programms.

Der Verfahrensablauf zum Herunterladen des On-Card-Anwendungsteils entspricht dem in **Fig.** 3.

Fig. 5 zeigt die erfinderischen Schritte zum Herunterladen von On-Card-Anwendungsteilen von einem Server über Netz auf eine Chipkarte in einer Java-Implementierung.

Hierbei soll davon ausgegangen werden, dass eine Brokerage-Anwendung, die auf einem Server abgelegt ist, auf die Chipkarte geladen werden soll. Auch Schlüssel zur Authentifizierung sind auf, dem Server abgelegt.

Der Client stellt eine Kommunikation zur Chipkarte und zum Server her. Die Kommunikation zur Chipkarte wurde mittels OCF (OpenCardFramework) realisiert.

Der Client stellt einen Request an den Server, dass die Brokerage-Anwendung (On-Card-Anwendungsteil) auf die Chipkarte gebracht werden soll. Client und Server kommunizieren vorzugsweise über TCP/IP oder HTTP.

Der Server sendet eine Response an den Client, mit der Aufforderung, die Chipkartenidentifikationsdaten (holeKartenInfo) zu senden.

Der Client empfängt die Response vom Server und sendet geeignete Kommando-APPUs an die Chipkarte, um die Chipkartenidentifikationsdaten zu erhalten. Die Chipkartenidentifikationsdaten sind im nichtflüchtigen der Chipkarte abgelegt und können durch geeignete Kommandos ausgelesen werden. Die Chipkarte empfängt die Kommandos und gibt die Chipkartenidentifikationsdaten an den Client zurück. Der Client sendet diese Daten in einem Request an den Server.

Der Server empfängt den Request und wertet die Chipkartenidentifikationsdaten aus, um herauszufinden, um welchen Kartentyp es sich handelt. Abhängig von diesem Kartentyp wird eine Methode zur Authentifizierung ausgewählt. In der vorliegenden Implementierung ist der Kartentyp eine VISA Open Platform Karte mit symmetrischen Schlüsseln. Der erste Schritt der Authentifizierung besteht darin, dass der Server eine Zufallszahl generiert und einen Schlüsselnummer auswählt und diese Informationen in einem Kommando verpackt und an den Client sendet. Der Client extrahiert das OCF Kommando und sendet dieses an das OCF Interface auf dem Client Rechner. Das OCF Interface wandelt das OCF Kommando in eine oder mehrere APDUs um und sendet diese an die Chipkarte. Die Chipkarte empfängt die APDUs, stellt fest, dass es sich um eine Authentifizierung-Kommando handelt, generiert eine Zufallszahl, und bildet aus beiden Zufallszahlen und dem mitgeteilten Schlüssel einen Session Key (Schlüssel) und sendet die Zufallszahlen damit verschlüsselt zurück.

Der Client gibt die Antwort der Karte an den Server. Der Server generiert aus den beiden Zufallszahlen und der Schlüsselnummer ebenfalls einen SessionKey. Mit Hilfe dieses SessionKeys überprüft er die verschlüsselten Zufallszahlen. Bei erfolgreicher Überprüfung wird die Karte als authentifiziert angesehen.

Der Server sendet ein zweites Authentifizierungskommando an den Client, um sich nach dem gleichen Verfahren – wie vorher beschrieben – zu authentifizieren. Bei erfolgreicher Überprüfung wird der Server als authentifiziert angesehen.

Die Brokerage-Anwendung wird auf dem Server mittels der SessionKeys signiert und gegebenenfalls verschlüsselt, um die Brokeranwendung herunterladen zu können. Diese Kommando-Sequenz bewirkt auf der Chipkarte die Erzeugung der Brokerage-Anwendung A. Die Kommando-Sequenz ist eine vordefinierte Sequenz, die nichtflüchtigen Speicherbereich des Servers abgelegt ist. Eine weitere Ausführungsform ist, dass die Kommando-Sequenz ganz oder teilweise mit Hilfe eines Programms auf dem Server erzeugt wird. Dies wird vorzugsweise dann angewendet werden, wenn mit Hilfe der Kommando-Sequenz auch kartenindividuelle Daten mit in den On-Card-Anwendungsteil integriert werden sollen.

Vorzugsweise wird jedes Kommando innerhalb der Sequenz mit Hilfe des Schlüssels signiert und gegebenenfalls verschlüsselt. Dies kann zum Beispiel dadurch erfolgen, dass das erste Kommando innerhalb der Sequenz mit Hilfe der Zufallszahl und des richtigen Schlüssels mit einem MAC (message authentification code) versehen wird und alle nachfolgenden Kommandos mit Hilfe des MACs des vorangehenden Kommandos und des richtigen Schlüssels mit einem MAC versehen werden. Die Sequenz mit den signierten und gegebenenfalls verschlüsselten Kommandos wird zum Client gesendet.

Der Client empfängt die Response mit der Kommando-Sequenz und sendet die Kommandos nacheinander zur Chipkarte. Die Chipkarte überprüft die Signatur und führt die Kommandos nur bei Richtigkeit aus.

Die erläuterten Verfahrensschritte können auch zur Personalisierung der neuen Anwendung/Brokerage-Anwendung dienen.

Patentansprüche

1. Verfahren zum Herunterladen von Anwendungsteilen von einem Server über ein Client auf eine Chipkarte, wobei der Server und der Client über ein verteiltes System miteinander verbunden sind, **gekennzeichnet durch** folgende Schritte:
   a) Bereitstellen eines geheimen Schlüssels oder SessionKeys durch den Server
   b) Laden einer Kommando-Sequenz von Befehlen zum Herunterladen des Anwendungsteils auf die Chipkarte
   c) Generieren einer digitalen Signatur mit dem geheimen Schlüssel oder SessionKey über jedes Kommando innerhalb der Kommando-Sequenz
   d) Senden der signierten Kommando-Sequenz als Datenpaket an den Client
   e) Entpacken des Datenpakets und Senden der einzelnen Kommandos der Sequenz nacheinander zur Chipkarte
   f) Überprüfen der digitalen Signatur der einzelnen Kommandos und Ausführung der Kommandos bei Richtigkeit der digitalen Signatur.
2. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass das Authentifizierungsverfahrerv zur Generierung des SessionKeys durch folgende Schritte ausgewählt wird:
   a) Senden einer Anfrage vom Server über den Client an die Chipkarte zur Übermittlung der auf der Chipkarte abgelegten Chipkartenidentifikationsdaten
   b) Auslesen der Chipkartenidentifikationsdaten aus dem nichtflüchtigen Speicher der Chipkarte und Übermittlung der Chipkartenidentifikationsdaten über den Client zum Server
   c) Ermitteln aus den Chipkartenidentifikationsdaten ein Authentifizierungverfahren mit dessen Hilfe ein zwischen Server und Chipkarte vereinbarter SessionKey generiert werden kann.
3. Verfahren nach Anspruch 2, dadurch gekennzeichnet, dass der SessionKey durch ein Authentifizierungsverfahren mit folgenden Schritten bestimmt wird:
   a) Generieren einer Zufallszahl und Auswählen eines geheimen Schlüssels durch den Server
   b) Senden der Zufallszahl nach Schritt a) über den Client zur Chipkarte
   c) Generieren einer Zufallszahl durch die Chipkarte
   d) Bilden aus beiden Zufallszahlen und mitgeteilten Schlüssel einen SessionKey
   e) Senden der verschlüsselten Zufallszahlen und der von der Chipkarte generierten Zufallszahl zum Server
   f) Generieren eines SessionKeys durch den Server und Überprüfen der verschlüsselten Zufallszahlen mit Hilfe des SessionKeys.
4. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass das verteilte System das Intranet oder Internet ist.
5. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass die Kommunikation zwischen Server und Client über SSL (Secure Sockets Layer) als Übertragungsprotokoll erfolgt.
6. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass auf dem Server zur Laufzeit ein Programm existiert, das mit dem Client kommuniziert und bei Bedarf die dem Server zugänglichen Schlüssel verwendet und das Protokoll festlegt, wann welche Nachrichten mit dem Client ausgetauscht und wann welche Schlüssel verwendet werden müssen, und dass auf dem Client zur Laufzeit ein Programm existiert, das sowohl mit der Chipkarte als auch dem Server kommuniziert und das Protokoll implementiert, das festlegt, wann welche Nachrichten mit der Chipkarte und dem Server ausgetauscht werden müssen.
7. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass die Chipkartenidentifikationsdaten zumindest aus einer Chipkartenserienummer und einem Chipkartentyp bestehen.
8. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass die digitale Signatur über einen symmetrischen Kryptoalgorithmus mit Hilfe des SessionKeys, der zwischen Client und Server vereinbart wird, oder über einen asymmetrischen Kryptoalgorithmus mit Hilfe eines privaten Schlüssels erfolgt, der auf der Chipkarte liegt, wobei der Server im Besitz des öffentlichen Schlüssels ist.
9. Verfahren nach Anspruch 8, dadurch gekennzeichnet, dass der symmetrische Kryptoalgorithmus DES oder Trible-DES und der asymmetrische Kryptoalgorithmus RSA, DSA oder ein Eliptic-Curve-Algorithmus.
10. Verfahren nach Anspruch 3, dadurch gekennzeichnet, dass mit Hilfe der Chipkartenidentifikationsdaten und des MasterKeys der geheime Schlüssel abgeleitet wird.
11. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass die Kommando-Sequenz zumindest aus einem Install-Befehl, ein oder mehreren Load-Befehlen

und einem abschließenden Install-Befehl besteht und in einer APDU-Struktur niedergelegt ist.

12. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass mittels des SessionKeys jedes Kommando innerhalb der Kommando-Sequenz verschlüsselt wird.

13. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass die Kommando-Sequenz eine vordefinierte Sequenz für eine bestimmte Anwendung ist, die auf dem nichtflüchtigen Speicher des Servers gespeichert ist und zur Laufzeit in den flüchtigen Speicher des Servers geladen werden.

14. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass die Kommando-Sequenz durch das Server-Programm nach Anspruch 6 zur Laufzeit generiert wird.

15. Verfahren nach Anspruch 14, dadurch gekennzeichnet, dass in die Kommando-Sequenz kartenindividuelle Daten eingebunden werden.

16. Verfahren nach Anspruch 13, dadurch gekennzeichnet, dass das erste Kommando innerhalb der Sequenz mit Hilfe der Zufallszahl und des geheimen Schlüssels mit einem MAC(message authentification code) versehen wird und alle nachfolgenden Kommandos mit Hilfe des MACs des vorangehenden Kommandos und des Schlüssels mit einem MAC versehen werden.

17. Vorrichtung, zumindest enthaltend folgende Komponenten:

a) Client zumindest enthaltend:
   aa) einen Browser
   bb) ein Computerprogrammprodukt zur Ausführung des Verfahrensschritts e) nach Anspruch 1
   cc) ein Lesegerät für die Chipkarte
b) Server enthaltend zumindest:
   aa) ein Computerprogrammprodukt zur Ausführung der Verfahrensschritte a) bis d) nach Anspruch 1
   bb) einen nichtflüchtigen Speicher zur Speicherung der geheimen Schlüssel bzw. des MasterSchlüssels
c) eine Kommunikationsverbindung zwischen Client und Server.

18. Client zumindest enthaltend:
a) ein Browser
b) ein Computerprogrammprodukt zur Ausführung des Verfahrensschritts e) nach Anspruch 1.

19. Client nach Anspruch 17 weiter enthaltend:
c) ein Chipkartenlesegerät
d) eine Chipkarte mit einem nichtflüchtigen Speicher zumindest enthaltend folgende Daten:
   aa) eine Kartennummer
   bb) ein Kartentyp
   cc) einen geheimen Schlüssel.

20. Computerprogrammprodukt, das im internen Speicher eines digitalen Rechners gespeichert ist, enthaltend Teile von Softwarecode zur Ausführung des Verfahrens nach Anspruch 1 bis 16, wenn das Produkt auf dem Rechner ausgeführt wird.

Hierzu 4 Seite(n) Zeichnungen

Nummer: DE 199 47 986 A1
Int. Cl.[7]: H 04 L 9/00
Offenlegungstag: 12. April 2001

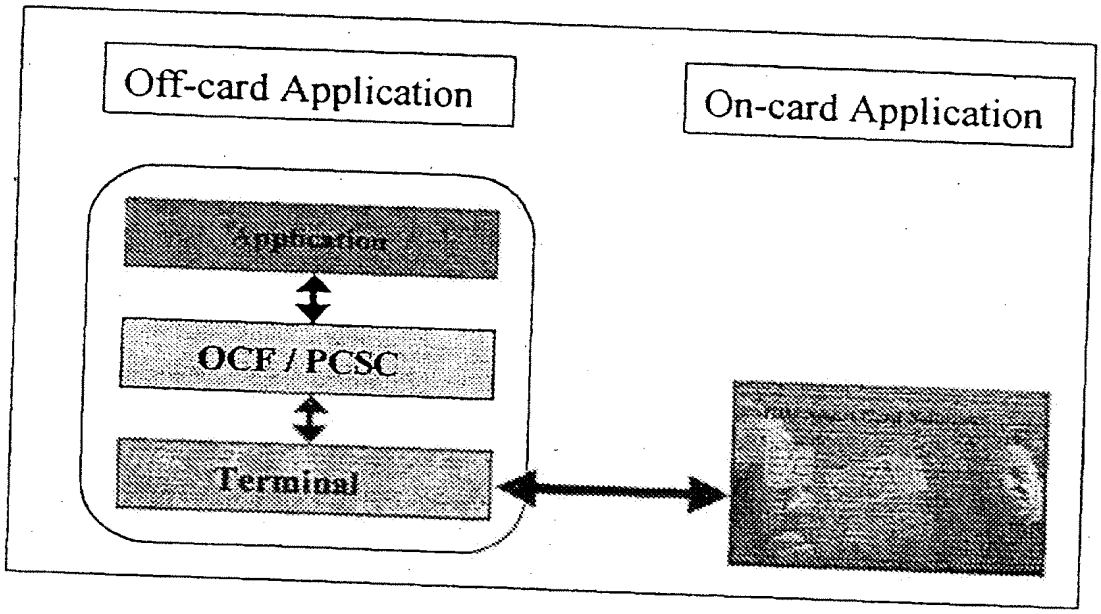
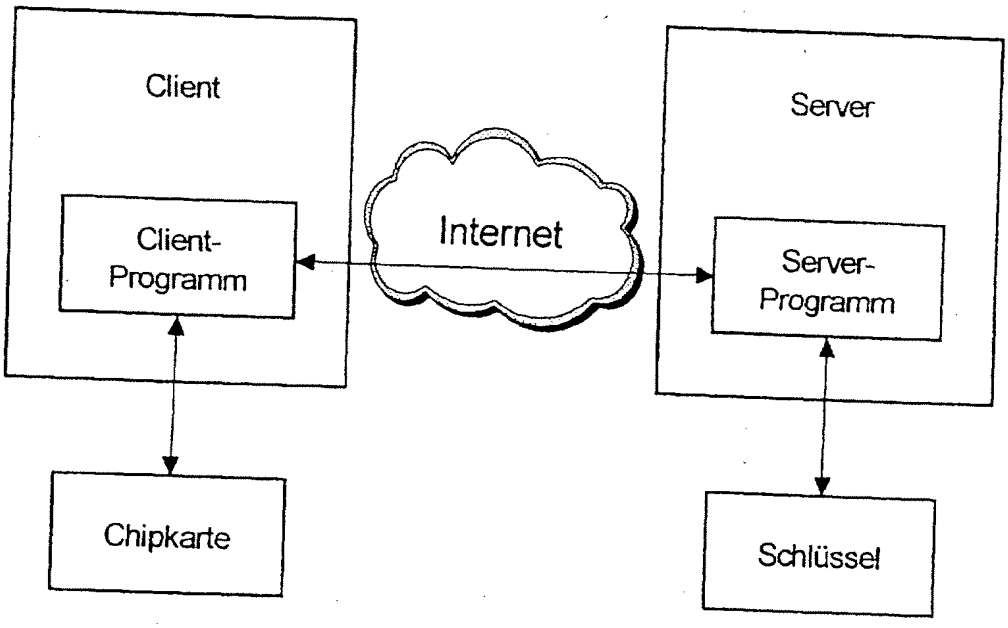


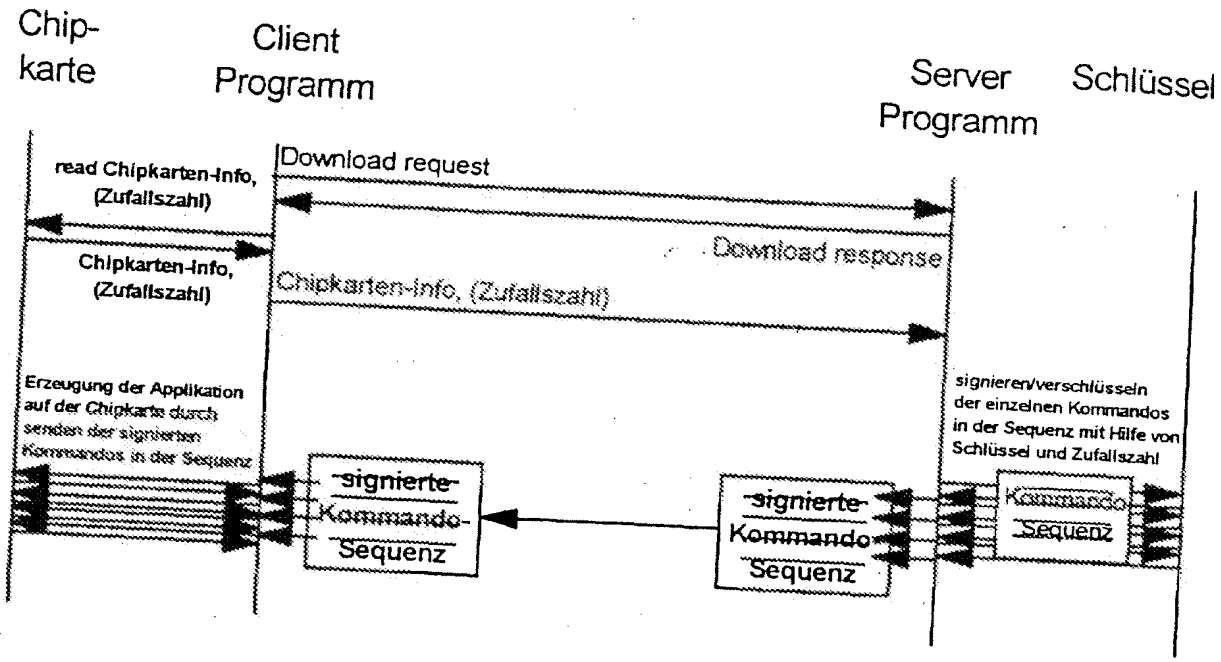
Fig. 1

Nummer: **DE 199 47 986 A1**
Int. Cl.[7]: **H 04 L 9/00**
Offenlegungstag: 12. April 2001

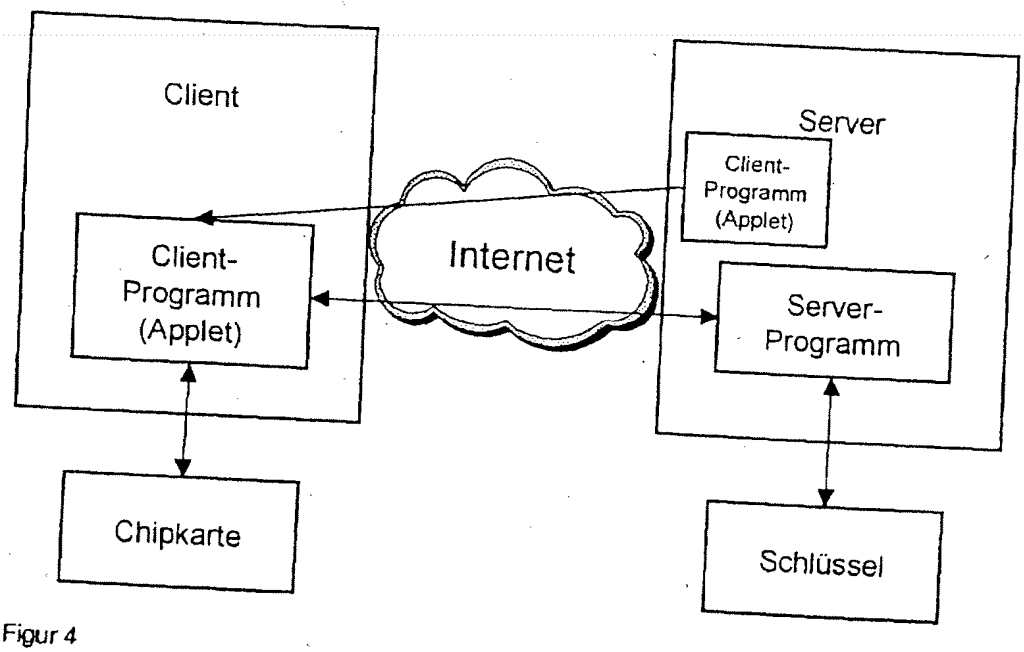


Figur 2

Nummer: DE 199 47 986 A1
Int. Cl.[7]: H 04 L 9/00
Offenlegungstag: 12. April 2001



Figur 3



Figur 4

Nummer: DE 199 47 986 A1
Int. Cl.[7]: H 04 L 9/00
Offenlegungstag: 12. April 2001

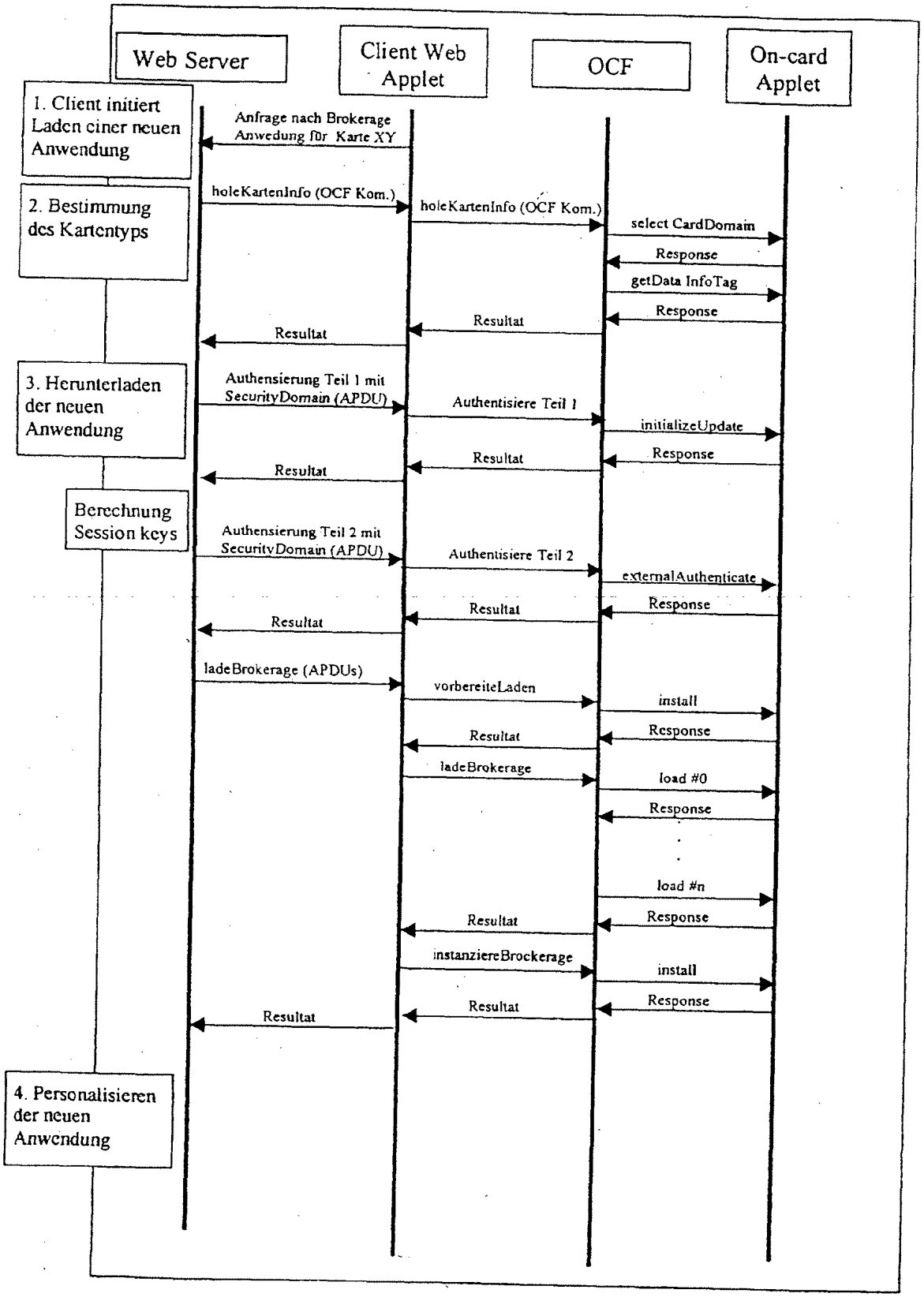


Fig. 5